

# Using a Tolerance-based Surrogate Method for Computer Resources Saving in Optimization

Martin MAREK, Petr KADLEC

Dept. of Radio Electronics, Faculty of Electrical Engineering and Communications, Brno University of Technology,  
Technická 10, 616 00 Brno, Czech Republic

marek@phd.feec.vutbr.cz, kadlec@feec.vutbr.cz

Submitted June 16, 2018 / Accepted January 9, 2019

**Abstract.** *This paper presents a very simple surrogate optimization method - a Tolerance-based Surrogate Method. A surrogate optimization in general is essential to more and more frequently used optimization in the development process of new technologies. Fitness functions of such systems are often costly, therefore keeping a number of evaluations of the fitness functions at minimum is of a great importance in order to save computer and time resources, i.e. the overall cost of design. Unlike other complex surrogate optimization methods, the tolerance-based surrogate method does not require excessive computational resources, is easy to implement, and is flexible for all types of optimization algorithms. Behaviour of the tolerance-based surrogate method is demonstrated on several modified benchmark problems. Afterwards, our method is verified on a real-world time-demanding optimization task.*

## Keywords

Optimization, surrogate optimization, tolerance-based surrogate method, resources saving

## 1. Introduction

Global optimization algorithms works so that they search for and compare different solutions to find the optimal one. The comparison of solutions is based on fitness values, which express the quality of the solution. The fitness values are obtained by evaluating fitness functions, which describe the behaviour of the optimized system with its properties called decision variables. Therefore, the optimization is a process of finding minima or maxima of the fitness values.

If the system is described by one fitness function, the optimization is called single-objective and a single decision space vector is expected as an optimization result. Contrarily, multiple conflicting fitness functions lead to the multi-objective optimization and multiple trade-off solutions are given at the end of the process.

The fitness functions can have various forms. If the fitness function is expressed as a closed form formula, it can be computed almost immediately. However, the fitness functions in real-world optimization problems can take considerably more time to compute. A common assumption is that the computation of the fitness values is the most time demanding operation during the optimization process.

An example of such a complex optimization can be the synthesis of the cavity resonator structure used in [1]. An optimization algorithm generates decision variables (e.g. design dimensions) and the calculation of the fitness values involves full-wave simulation of the designed structure with dimensions determined by the optimization algorithm.

Since evolutionary algorithms generally require a large number of fitness function evaluations during the optimization process and each evaluation can take a significant amount of time, it is desirable to be able to skip some unnecessary fitness functions evaluations.

There are various methods described in an open literature dealing with "redundant" fitness functions evaluations, which are in general called surrogate optimization methods. In [2], response surfaces are used to approximate the fitness functions, that are evaluated only at a few points. Authors in [3] proposes Progressive Optimum Search Using Evolving Reliable Regions (POSER) method, which at first establishes a Kriging model (contains an error prediction) from a few initial samples and then applies it to create a reliable region. The reliable region of the Kriging surrogate is progressively improved by additional samples. The method presented in [4] reduces the number of fitness function evaluations by fitting a function approximation model over  $k$  nearest previously evaluated points. The method tries to identify the most promising offspring solutions and exploit their potential before other offspring solutions. The approximation model uses Symmetric Latin Hypercube Design (SLHD). In [5], a trust-region framework using an interpolating Radial Basis Function (RBF) model is used. The surrogate optimization methods were summarized in [6].

All the methods proposed in the open literature approximate or interpolate the unknown regions of the fitness function by sampling it by minimal possible points to obtain a reliable surrogate. Such complex methods are undoubtedly able to estimate more accurate substitute solution than our proposed method. Nevertheless the tolerance-based surrogate method simply stores all the evaluated solutions in an archive and uses the stored fitness values later if a new solution is within a specified margin. It can be used on any problem with any single-objective or multi-objective algorithm. While the methods proposed in [2–6], and other surrogate optimization methods to be found in the open literature, are rather difficult to implement, our tolerance-based surrogate method is, for its simple principle, very easy to implement and is also universal for all kinds of optimization algorithms and problems. Therefore, it has a potential to help many engineers in various engineering branches in their efforts to reduce design cost without deep studying of the problem and implementing the complex surrogate methods.

In Sec. 2, an optimization technique used for the validation of the proposed method is described. In Sec. 3, the principle of the tolerance-based surrogate method is described. In Sec 4 and 5, metrics used for the validation of the results and problems for benchmarking the results are discussed. Section 6 is dedicated to the experimental verification of the proposed method. Section 7 presents the use of the tolerance-based surrogate method for the design of a band-stop filter. Finally, the conclusion of the paper is given in Sec. 8.

## 2. Optimization Technique

For the validation purposes of this paper the FOPS toolbox [7] was used. The Multi-objective Particle Swarm Optimization (MOPSO) [8] algorithm has been exploited to obtain presented results. The Elitist Non-dominated Sorting Genetic Algorithm (NSGA-II) [9] and the Third Generalized Differential Evolution algorithm (GDE3) [10] were also tried and the produced results were practically similar to those from MOPSO algorithm.

Minor differences in the results were related to algorithms' performance rather than to the tolerance-based surrogate method itself. Therefore, the results from the GDE3 algorithm are not presented in this paper. Although the results of the NSGA-II algorithm are also similar to those of MOPSO algorithm, they are attached to the paper in an appendix 1, due to a completely different nature of the NSGA-II algorithm.

The MOPSO algorithm is based on the simulation of the social behaviour of bees in a swarm. The position of a particle is changed according to its own experience and that of its neighbours according to equation [8]:

$$\mathbf{x}_t = \mathbf{x}_{t-1} + \Delta t \cdot \mathbf{v}_t \quad (1)$$

where  $\mathbf{x}_t$  is the position of the particle at a time step  $t$ ,  $\Delta t$  is the time step ( $\Delta t = 1$ ), and the  $\mathbf{v}_t$  is a velocity vector at

the time step  $t$ . The velocity vector reflects the exchange of information and is defined as follows [8]:

$$\begin{aligned} \mathbf{v}_t = & w \cdot \mathbf{v}_{t-1} + c_1 \cdot r_1 \cdot (\mathbf{x}_{pbest} - \mathbf{x}_{t-1}) \\ & + c_2 \cdot r_2 \cdot (\mathbf{x}_{gbest} - \mathbf{x}_{t-1}) \end{aligned} \quad (2)$$

where  $w$  is the inertia weight,  $c_1$  and  $c_2$  are cognitive and social learning factors, respectively,  $r_1$  and  $r_2 \in [0, 1]$  are random values,  $\mathbf{x}_{pbest}$  is the position of a personal best, and  $\mathbf{x}_{gbest}$  is the position of a global best.

A multi-objective variant of PSO algorithm is extended by an external archive, which is the container that stores non-dominated solutions found during the optimization run. Therefore, the global best solution ( $\mathbf{x}_{gbest}$  in (2)) is selected among external archive members. To avoid overloading of the external archive, a pruning method based on a crowding distance [9] is used.

## 3. Tolerance-based Surrogate Method

As was mentioned before, the tolerance-based surrogate method allows an optimization algorithm to skip some fitness function evaluations. The question is, which evaluations can be skipped?

If an electromagnetic structure design is considered, there are always some manufacture precision limits, therefore it is useless to evaluate the fitness values for dimensions (decision variables) that differ at e.g. sixth decimal place. Moreover, the fitness values of such similar dimensions would most likely be very similar too and an overall contribution to the optimization process would be minimal.

This is the essential idea of the tolerance-based surrogate method. At the beginning of the optimization run, no fitness values are known and all the fitness function has to be evaluated. Each evaluated solution (i.e. decision variables and corresponding fitness values) is stored in the archive. At some point in the optimization process, an algorithm converges close to the true global optimum (minimum or maximum for single-objective optimization and Pareto-front for multi-objective optimization) and new solutions with yet unknown fitness values begin to be similar (or equal) to some members of the archive. Evaluation of the fitness functions of such solution has a negligible contribution to the optimization process, therefore the fitness functions are not evaluated and the fitness values of the closest solution in the archive are taken from the archive.

How close the new solution from a member of the archive has to be, is defined by the vector of tolerances, which has a number of elements equal to a number of decision variables of a problem. Each time the differences between all the decision variables of some member of the archive and the new solution are lower than the vector of tolerances, the fitness function evaluation is skipped.

Figure 1 further clarifies the tolerance-based surrogate method. It depicts a decision space of a simple two-objective optimization problem and several solutions stored in the archive. A light grey grid denotes the limits of the decision variables, i.e.  $x_1 \in \langle 0.1, 1 \rangle$  and  $x_2 \in \langle 0, 1 \rangle$ . The fitness functions are defined as follows:

$$f_1(\mathbf{x}) = x_1, \quad (3)$$

$$f_2(\mathbf{x}) = \frac{1 + x_2}{x_1}. \quad (4)$$

A red line marks the true Pareto-front of the problem in decision space. There are 15 solutions stored in the archive (their positions are marked with the black thick crosses). The tolerance vector was set to  $\{0.05, 0.1\}$  and areas within tolerance are marked by the hatched boxes around solutions. Five of the solutions are marked with the index number from **1** to **5**.

The solution **1** is the true Pareto-optimal solution and its fitness values are  $\{0.1, 10\}$ . The solution **2** has the fitness values  $\{0.3, 3.67\}$ . The solution **3** is not far from optimality (see that a tolerance box covers a part of the true Pareto-front) and its fitness values are  $\{0.5, 2.15\}$ . The solution **4** has the fitness values  $\{0.7, 1.643\}$  and the solution **5** has the fitness values  $\{1, 1\}$  (also the true Pareto-optimal solution). All the indexed solutions are non-dominated (in objective space).

If a newly generated solution has the position e.g.  $\{0.94, 0\}$  (marked with the blue thin cross in Fig. 1), it will fall in the tolerance area of the solution **5** ( $\{1, 0\}$ ) and even if its fitness values according to (3) and (4) would be  $\{0.94, 1.064\}$ , the fitness values  $\{1, 1\}$  of the solution **5** will be assigned to it. A deviation in fitness values caused by tolerance-based surrogate method is relatively small in this case.

Another generated solution has the position e.g.  $\{0.14, 0\}$  (marked with the blue thin cross in Fig. 1). Such solution will fall in the tolerance area of the solution **1** ( $\{0.1, 0\}$ ) and even if its fitness values according to (3) and (4) would be  $\{0.1, 7.143\}$ , the fitness values  $\{0.1, 10\}$  of the solution **1** will be assigned to it. The difference between the true fitness values and the surrogate fitness values is rather large here, although the absolute distance between the archive member and the generated solution in the decision space is identical as in the case described in the previous paragraph. This suggests that the setting of the tolerance vector can be sometimes a difficult task.

The solution **3** in Fig. 1 indicates the main drawback of the tolerance-based surrogate method. The border of the hatched box of this solution lies on the true Pareto-front, but the solution itself is rather far away ( $\{0.5, 0.075\}$ ). Therefore, if a new solution is generated within the hatched box, e.g.  $\{0.5, 0\}$  (marked with the blue thin cross), then the fitness values of known solution are assigned to it. But the fitness values of the true Pareto-optimal solution with the position  $\{0.5, 0\}$  according to equations (3) and (4) are  $\{0.5, 2\}$ ,

while the fitness values of the solution **3** from the archive are  $\{0.5, 3.5\}$ . Afterwards, the solution  $\{0.5, 0\}$  (which is, as we know, better then the archive member) will be suppressed in the optimization process due to its downgraded fitness values. This denotes that a part of the true Pareto-front under the solution **3** in Fig. 1 is inaccessible due to the tolerance-based surrogate method when too large values are used in the tolerance vector.

There exists no methodology to estimate the proper tolerance vector. The tolerance vector depends on an optimized problem and user's knowledge about the problem. This drawback of the tolerance-based surrogate method that can make the parts of the true Pareto-front inaccessible, and therefore introduces an uncertainty into the optimization process, is balanced by the positive effect on the overall number of fitness evaluations i.e. overall cost of optimization. In other words, the setting of the tolerance vector is a trade-off between the time saving properties and the inaccessible area that might occur around the true global optimum.

Note that an optimization algorithm can still reach any point within the decision space. It can not reach only a close neighbourhood of the archive members.

The drawback can be suppressed with the use of a discrete decision space. When the decision space is discrete, the tolerance vector can be set to almost zero values and a new solution can be either identical or differ by a step of the discrete decision variable. If the new solution generated by the optimization algorithm already exists in the archive, it is not calculated again. In this scenario, some regions of the decision space are inaccessible for the optimization algorithm.

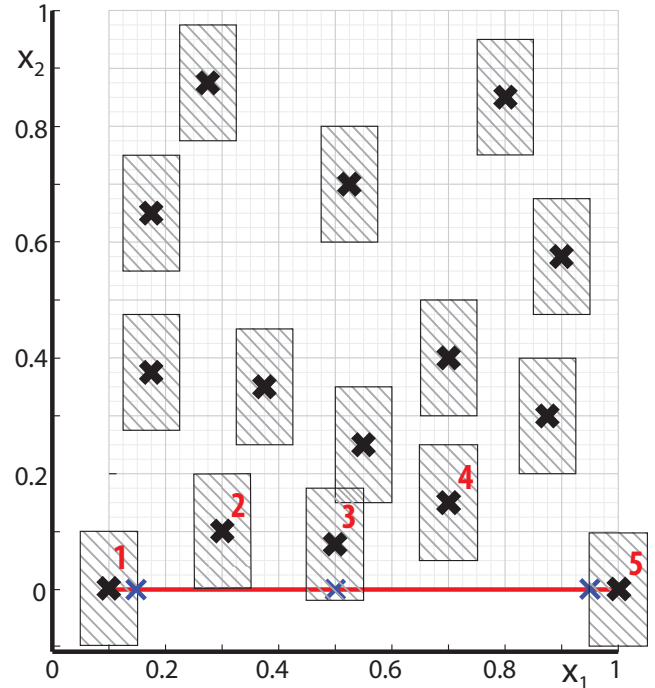


Fig. 1. Decision space of a two-objective problem with solutions stored in the archive.

## 4. Evaluated Metrics

The performance of the tolerance-based surrogate method is tested from a two points of view. The first one is the computational time required to perform the particular simulation run. For a better insight into the time saving property of the tolerance-based surrogate method, Table 3 contains an average count of fitness values obtained by our surrogate method.

The second point of view is the value of a generational distance. The generational distance was proposed in [11]. It defines the distance between a non-dominated set  $P$  and a true Pareto-front  $P^*$ . It is obtained by the equation:

$$GD = \frac{\sqrt{\sum_i^{|P|} d_i^2}}{|P|} \quad (5)$$

where  $d_i^2$  is the minimal Euclidean distance in an objective space between  $i$ -th solution from the set  $P$  and any member of the true Pareto-front  $P^*$ :

$$d_i^2 = \min_{k=1}^{|P^*|} \sqrt{\sum_{m=1}^M [f_m(i) - f_m^*(k)]^2} \quad (6)$$

where  $f_m^*(k)$  is the  $m$ -th fitness value of  $k$ -th member from the set  $P^*$ .

## 5. Testing Problems

The validation of the tolerance-based surrogate method was performed on numerous two-objective optimization benchmark problems. However, the tolerance-based surrogate method is independent on the number of objectives.

A summary of benchmark problems can be seen in Tab. 1 (MOFON stands for Fonseca and Fleming's study, MOKUR stands for Kursawe's study, MOPOL stands for Poloni's study and MOZDT1 and MOZDT6 stands for Zitzler, Deb and Thiele's studies). The benchmark problems are further described in [12].

The generational distance metric uses the true Pareto-fronts  $P^*$  for the distance calculation. The true Pareto-fronts of MOFON, MOZDT1, and MOZDT6 problems can be found in [12], while the true Pareto-fronts of MOPOL and MOKUR problems were obtained thanks to a very dense sampling of the regions, where the true Pareto-front is located.

## 6. Results

Controlling parameters of the MOPSO algorithm were set as follows: the inertia weight  $w$  was linearly decreased from 0.6 to 0.4 over each iteration, the cognitive learning factor was  $c_1 = 1.5$ , and the social learning factor was  $c_2 = 1$ .

There were 100 agents in each simulation run over 100 iterations. Therefore, the fitness function would be evaluated 10 000-times if the tolerance-based surrogate method was disabled.

An evaluation of the fitness function in case of the benchmark problems is almost immediate, therefore the usage of the tolerance-based surrogate method would have no benefit. Nevertheless, there were delays inserted to the fitness functions. The delays were 0, 1, and 10 milliseconds. Due to the nested delays, all evaluations of the fitness functions alone took 0, 10, and 100 seconds, respectively, for each simulation run if the tolerance-based surrogate method was disabled.

The tolerance vector was defined as a fraction of the range of problem's decision variables, i.e. 0, 0.001, 0.01, 0.05, and 0.1 times the range of the decision variable. The first one means that the range of each decision variable is "divided" into an infinite number of sections. In other words, the tolerance-based surrogate method is disabled. The last one means, that the range of each decision variable is "divided" into 10 sections. The quotation marks refer to the fact, that the tolerance-based surrogate method has nothing to do with the discretization of the decision variables. The tolerance-based surrogate method only takes positions of two randomly generated solutions and checks whether its difference is lower than the tolerance or not. All values presented in Tabs. 2–5 are an average of 100 repetitions.

### 6.1 Computational Time

Table 2 contains an average computational time of particular simulations. It is obvious from the first three lines (the tolerance-based surrogate method is disabled), that the computational time of an optimization method alone is almost independent on a problem. It is also evident how the nested delays affect the computational times.

Following lines show the computational times when the tolerance-based surrogate method is enabled. The higher the tolerance is, the larger the time saving is. Contrarily, the computational time when the tolerance-based surrogate method is enabled depends on the problem.

On several occasions, the computational time is larger when the tolerance-based surrogate method is enabled, compared to the computational times with the tolerance vector elements set to zero. The most obvious items are the ones where no delay and small tolerance values were used. This behaviour is caused by a number of comparison operations required by the tolerance-based surrogate method. If the tolerance is small and no surrogate fitness values are found in the archive (MOKUR and MOZDT1 problems, see Tab. 3), the number of the comparison operations quickly increases during the optimization which slows the entire process.

Especially, in case of the MOZDT1 problem and the tolerance vector elements set to 0.001, no surrogate solutions were found (see Tab. 3). This problem has 30 decision variables, therefore the probability that a new solution is within the tolerance of some solution stored in the archive is lower compared to other problems. The number of the comparison operations quickly grows from  $100 \times 30$  after first iteration to  $10000 \times 30$  after last iteration. Therefore, an overall deceleration is almost 13 seconds.

Problem	MOFON	MOKUR	MOPOL	MOZDT1	MOZDT4
Number of decision variables	3	3	2	30	10
Limits of decision variables	$\langle -4, 4 \rangle$	$\langle -5, 5 \rangle$	$\langle -\pi, \pi \rangle$	$\langle 0, 1 \rangle$	$\langle -5, 5 \rangle^*$

\* The first decision variable has limits  $\langle 0, 1 \rangle$ .

**Tab. 1.** Summary of benchmark problems used in the paper.

Delay [ms]	Tolerance	MOFON	MOKUR	MOPOL	MOZDT1	MOZDT4
0	0	2.95	2.86	2.62	2.74	2.68
1	0	13.66	13.75	12.82	13.36	13.13
10	0	103.42	103.11	103.09	103.10	103.10
0	0.001	5.07	5.26	4.53	16.61	5.18
1	0.001	10.94	15.36	11.14	27.76	13.21
10	0.001	58.19	100.36	65.50	117.85	78.53
0	0.01	3.79	4.13	3.79	17.09	5.02
1	0.01	5.40	6.63	6.14	27.84	11.00
10	0.01	17.97	24.97	22.78	116.15	54.68
0	0.05	2.98	3.16	2.95	14.78	4.60
1	0.05	3.48	3.76	3.26	21.69	8.04
10	0.05	7.37	8.47	5.39	74.52	34.11
0	0.1	2.81	2.91	2.70	6.00	4.30
1	0.1	3.04	3.20	2.83	10.99	6.60
10	0.1	4.81	5.24	3.84	45.97	23.53

**Tab. 2.** An average computational time in seconds.

Delay [ms]	Tolerance	MOFON	MOKUR	MOPOL	MOZDT1	MOZDT4
0	0.001	4777	597	4024	0	2870
1	0.001	4812	596	4045	0	2755
10	0.001	4775	587	3992	0	2751
0	0.01	8620	7991	8177	230	4788
1	0.01	8626	7976	8171	228	4801
10	0.01	8624	7980	8170	199	5117
0	0.05	9576	9488	9778	4014	7080
1	0.05	9575	9497	9777	3970	7160
10	0.05	9574	9493	9775	4101	7135
0	0.1	9805	9774	9889	6159	8192
1	0.1	9806	9775	9889	6148	8201
10	0.1	9806	9779	9889	6128	8152

**Tab. 3.** An average number of surrogate solutions.

Delay [ms]	Tolerance	MOFON	MOKUR	MOPOL	MOZDT1	MOZDT4
0	0	<0.001	0.032	0.011	0.247	20.822
1	0	<0.001	0.032	0.012	0.232	20.649
10	0	<0.001	0.032	0.013	0.221	19.965
0	0.001	<0.001	0.033	0.012	0.234	23.354
1	0.001	<0.001	0.031	0.012	0.245	22.911
10	0.001	<0.001	0.034	0.010	0.225	23.507
0	0.01	0.005	0.110	0.037	0.226	46.445
1	0.01	0.005	0.109	0.036	0.238	49.949
10	0.01	0.005	0.111	0.039	0.232	47.377
0	0.05	0.051	0.893	0.322	0.261	61.551
1	0.05	0.052	0.869	0.306	0.262	60.981
10	0.05	0.052	0.888	0.313	0.259	60.895
0	0.1	0.129	2.153	0.903	0.337	70.144
1	0.1	0.130	2.068	0.929	0.334	68.461
10	0.1	0.123	2.064	0.965	0.335	68.306

**Tab. 4.** An average generational distance - Real-coded decision space.

Delay [ms]	Fraction	MOFON	MOKUR	MOPOL	MOZDT1	MOZDT4
0	0	<0.001	0.032	0.011	0.235	21.542
1	0	<0.001	0.033	0.010	0.231	19.827
10	0	<0.001	0.032	0.014	0.242	17.982
0	0.001	<0.001	0.030	0.011	0.110	11.954
1	0.001	<0.001	0.030	0.011	0.111	14.233
10	0.001	<0.001	0.030	0.011	0.129	12.823
0	0.01	0.001	0.057	0.027	0.055	11.033
1	0.01	0.001	0.058	0.027	0.043	11.574
10	0.01	0.001	0.057	0.027	0.041	12.019
0	0.05	0.026	0.219	0.277	0.041	0.145
1	0.05	0.026	0.219	0.311	0.045	0.146
10	0.05	0.026	0.219	0.311	0.038	0.146
0	0.1	0.102	0.017	0.288	0.074	0.164
1	0.1	0.102	0.017	0.288	0.046	0.167
10	0.1	0.102	0.017	0.288	0.063	0.160

**Tab. 5.** An average generational distance - Discrete decision space.

## 6.2 Number of Surrogate Solutions

Table 3 shows, how many fitness values were taken from the archive during each simulation run. The content of Tab. 3 correlates with the content of Tab. 2. The first three lines of the table were omitted, because they obviously contain zeros because the tolerance-based surrogate method is disabled. With an increasing tolerance values the number of surrogate solutions also increases, because there is a higher probability of finding a close enough solution in the archive.

The differences between values with the same tolerance but different problems are firstly set by a number of decision variables. If a problem has only three decision variables (MOFON), it is much easier to randomly generate a close solution to a member of the archive compared to a problem with 30 decision variables (MOZDT1).

Secondly, the differences are given by the speed of convergence to an optimum. If agents rapidly approach global optimum, then new solutions are almost similar to the previous ones, therefore surrogate solutions can be found in the archive (MOFON). Contrarily, if agents approach the optimum slowly, the positions are continuously drawn to optimum (MOKUR), and the surrogate solutions cannot be found in the archive.

## 6.3 Generational Distance

When the tolerances are increased, the generational distance also increases due to the drawback of the tolerance-based surrogate method described in Sec. 3. Differences between values of the generational distance with the same tolerance are caused by the difficulty of the problem. The MOFON problem is relatively simple one. On the other hand, the MOZDT4 problem with only 10 decision variables (in comparison with the MOZDT1 problem) has many local optima, where an algorithm can be caught, therefore the values of the generational distance are large.

Table 5 contains generational distance values of simulations when a discrete decision space was used. Note that the second column in Tab. 5 is now called Fraction. In this case, the tolerances were set to very low values ( $1e-6$ ). However, the discretization of the decision variables corresponds with the tolerances from the simulation with a real-coded decision space. Therefore, the fraction of 0.1 denotes that each decision variable was sampled by 11 points. When the discrete decision space is used, the surrogate is found only if a new solution is identical to a member of the archive. Otherwise, the fitness functions has to be evaluated.

Some problems (MOPOL and MOFON) in Tab. 5 show that if the fraction value is increased, meaning that the decision variable is sparsely sampled, the generational distance downgrades. This is caused by the fact that the true Pareto-optimal solutions do not correspond with the samples of the decision variables. The true Pareto-optimal set of MOZDT problems corresponds to  $x_1 \in \langle 0, 1 \rangle$  while all the other decision variables are zero. Therefore, the discrete samples of the decision variables can match the true Pareto-optimal set.

Analogous tables to Tab. 2 and Tab. 3 with the discrete decision space are not presented, because their content is similar to those with the real-coded decision space.

## 7. Anisotropic Band-Stop Filter Design

Until now, only artificial, benchmark problems were considered for the verification of the tolerance-based surrogate method in this paper. The benchmark problems were unnaturally altered by inserting delays to introduce reproducible results, but the use of the tolerance-based surrogate method on such optimization tasks is meaningless.

The tolerance-based surrogate method was advantageously exploited in the synthesis of the electromagnetic equivalents of composite sheets [13]. An anisotropic band-stop filter based on a microstrip line above a uniplanar band-gap (UBG) ground plane was designed by a multi-objective optimization.

The Anisotropic band-stop filter is formed by the microstrip line above a ground plane with an array of etched slots of varying widths (Fig. 2). By changing a number of step-impedance slot lines  $N$ , a slot period  $a$  (dimension of etched slots) and the angle  $\varphi$  between the microstrip line and the step-impedance slots, transmission properties of the filter are being changed.

The transmission characteristics of the band-stop filter were obtained by a full-wave analysis in the transient solver of CST Microwave Studio. The design properties  $N$ ,  $a$  and  $\varphi$  were acting as decision variables and fitness functions evaluation consists of a full-wave analysis in the CST Microwave Studio and parsing of the transmission characteristics to achieve the fitness values. An RT/Duroid substrate ( $h = 0.635$  mm,  $\epsilon_r = 10.2$ ,  $t = 35$   $\mu$ m) was used in the optimized structure.

It is obvious, that such evaluation is time demanding (approximately 5 to 30 minutes), therefore a great emphasis should be put on keeping the overall number of the fitness functions evaluations at minimum.

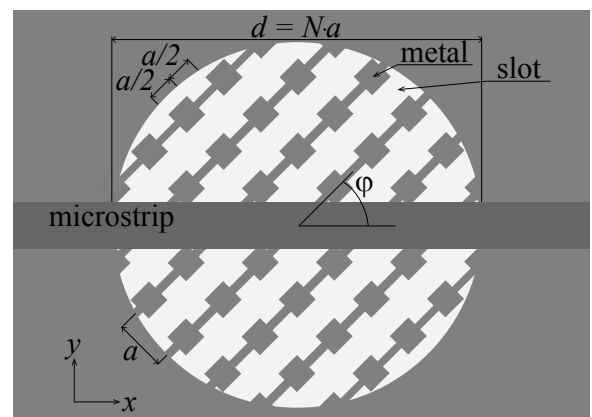


Fig. 2. UBG structure with state variables. Gray: metal strips, white: slots.

## 7.1 Optimization Parameters

Decision variables were discrete and they were defined as follows: the number of the step-impedance slot lines  $N \in [5, 7, 9]$ , the slot period  $a \in [0.5, 0.6, \dots, 2.0]$  mm and the angle between the microstrip line and the step-impedance slots  $\varphi \in [0, 2, 4, \dots, 90]^\circ$ . The fitness values were obtained from a frequency response of the transmission coefficient defined as follows (see Fig. 3):

$$|S_{21}(\mathbf{x}, F)| > -5 \text{ dB}, F \in \langle 0 \text{ GHz}; 4 \text{ GHz} \rangle, \quad (7)$$

$$|S_{21}(\mathbf{x}, F)| < -20 \text{ dB}, F \in \langle 5 \text{ GHz}; 8 \text{ GHz} \rangle, \quad (8)$$

$$|S_{21}(\mathbf{x}, F)| > -5 \text{ dB}, F \in \langle 9 \text{ GHz}; 10 \text{ GHz} \rangle \quad (9)$$

where  $\mathbf{x} = [a, N, \varphi]^T$  is the position of a solution and  $F$  denotes a frequency.

Each frequency band forms one fitness function defined by (10)–(12). Basically, it is a sum of  $S_{21}$  values that violates the defined frequency mask (7)–(9). The CST Microwave Studio produces 5001 frequency samples of  $S_{21}$  within the interval from 0 GHz to 12 GHz.

$$f_1(\mathbf{x}) = \sum \diamond [-S_{21}(\mathbf{x}, F) - 5], F \in \langle 0 \text{ GHz}; 4 \text{ GHz} \rangle, \quad (10)$$

$$f_2(\mathbf{x}) = \sum \diamond [20 + S_{21}(\mathbf{x}, F)], F \in \langle 5 \text{ GHz}; 8 \text{ GHz} \rangle, \quad (11)$$

$$f_3(\mathbf{x}) = \sum \diamond [-S_{21}(\mathbf{x}, F) - 5], F \in \langle 9 \text{ GHz}; 10 \text{ GHz} \rangle \quad (12)$$

where the operator  $\diamond$  denotes that an output is equal to an argument in square brackets only if the argument is positive. Otherwise, the output is zero.

## 7.2 Optimization

The paper [13] presents the comparison of the performance of two algorithms, GDE3 and NSGA-II, in design of the band-stop filter. Both algorithms had 20 agents over 20 iterations, which means that each algorithm needed 400 fitness function evaluations. To obtain independent realizations of stochastic processes, both algorithm runs were 10 times repeated.

In other words,  $2 \times 400 \times 10 = 8000$  fitness function evaluations would be normally needed. It would take almost 56 days to evaluate the fitness functions (assuming that each fitness function evaluation takes 10 minutes). However, an overall number of possible solutions is  $3 \times 16 \times 46 = 2208$  (the decision variables are discrete). This suggests that some solutions would certainly be evaluated more than once, which encourages to the use of the tolerance-based surrogate method.

Thanks to the tolerance-based surrogate method, the whole procedure took about 15 days and the total number of 2022 solution had to be full-wave analysed by the CST Microwave Studio (an average fitness function evaluation took little over 10 minutes). The remaining solutions (2208 overall) were not reached during any optimization run.

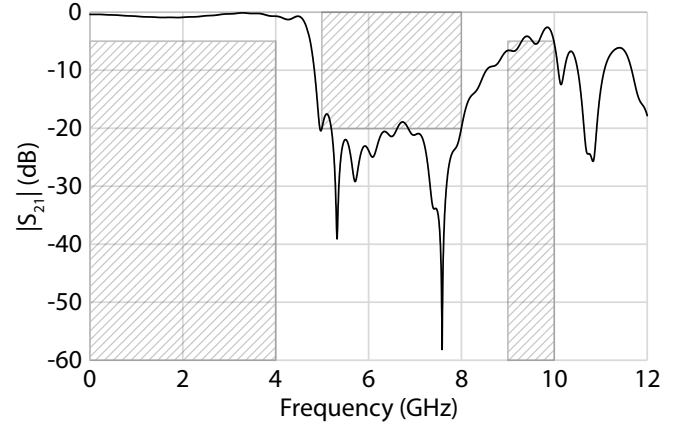


Fig. 3. Frequency response of transmission coefficients of the best solution found in optimization process.

Figure 3 shows the result of the anisotropic band-stop filter design using the multi-objective optimization. The figure also shows the intended frequency mask (hatched areas) described by equations (7)–(9).

## 8. Conclusion

The Tolerance-based Surrogate Method reducing the time of the optimization process has been introduced. It has been described, that certain fitness function evaluations are unnecessary to evaluate, therefore an overall computational time of an optimization process can be reduced.

The drawback of skipping the fitness function evaluation can lead to a loss of a precision. The precision loss might be reduced if a discrete decision space is used with an appropriate tolerance vector.

It was also discussed that if the tolerance-based surrogate method is improperly used, the optimization process can be even slowed.

The real-world optimization task, the anisotropic band-stop filter design, was also presented. The fitness function evaluation took around 10 minutes and the overall optimization time was reduced from around 2 months to approximately 2 weeks thanks to the tolerance-based surrogate method.

Since an evaluation of fitness functions can be very time consuming, the proposed tolerance-based surrogate method can accelerate the whole optimization process even if only few surrogate solution are found.

The tolerance-based surrogate method can also be exploited in cases of recurrent optimization tasks either after altering algorithm settings or the crash of a simulation, because the archive of known solutions can be inserted before the beginning of the optimization process and surrogate solutions can be used from early stages of the optimization process.



## Acknowledgments

Research was supported by the grant LO1401 of the National Sustainability Program. Infrastructure of the SIX Center was used.

## References

- [1] SEDENKA, V., RAID, Z. Critical comparison of multi-objective optimization methods: genetic algorithms versus swarm intelligence. *Radioengineering*, 2010, vol. 19, no. 3, p. 369–377. ISSN: 1210-2512
- [2] JONES, D. R., SCHONLAU, M., WELCH, W., J. Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, 1998, vol. 13, no. 4, p. 455–492. DOI: 10.1023/A:1008306431147
- [3] YU, J.-C., et al. Evolutionary algorithm using progressive Kriging model and dynamic reliable region for expensive optimization problems. In *International Conference on Systems, Man, and Cybernetics (SMC)*. Budapest (Hungary), 2016, p. 4383–4388. DOI: 10.1109/SMC.2016.7844920
- [4] REGIS, R. G., SHOEMAKER, C. A. Local function approximation in evolutionary algorithms for the optimization of costly functions. *IEEE Transactions on Evolutionary Computation*, 2004, vol. 8, no. 5, p. 490–505. DOI: 10.1109/TEVC.2004.835247
- [5] WILD, S. M., REGIS, R. G., SHOEMAKER, C. A. ORBIT: Optimization by radial basis function interpolation in trust-regions. *SIAM Journal on Scientific Computing*, 2008, vol. 30, no. 6, p. 3197–3219. DOI: 10.1137/070691814
- [6] JONES, D. R. A taxonomy of global optimization methods based on response surfaces. *Journal of Global Optimization*, 2001, vol. 21, no. 4, p. 345–383. DOI: 10.1023/A:1012771025575
- [7] MAREK, M., KADLEC, P., CUPAL, M. FOPS - Fast Optimization Procedures. *AToM - Antenna Toolbox for Matlab*, <http://antennatoolbox.com/fops-about.php>, 2017, [Online; accessed 6-January-2019]
- [8] REYES-SIERRA, M., COELLO, C., C. Multi-objective particle swarm optimizers: A survey of the state-of-the-art. *International Journal of Computational Intelligence Research*, 2006, vol. 2, no. 3, p. 287–308. DOI: 10.5019/j.ijcir.2006.68
- [9] DEB, K., PRATAP, A., AGARWAL, S., et al. A fast and elitist multi-objective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 2002, vol. 6, no. 2, p. 182–197. DOI: 10.1109/4235.996017
- [10] KUKKONEN, S., LAMPINEN, J. GDE3: The third evolution step of generalized differential evolution. In *Congress on Evolutionary Computation*, 2005, vol. 1, p. 443–450. DOI: 10.1109/CEC.2005.1554717
- [11] VELDHUIZEN, D. *Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations*. School of Engineering of the Air Force Institute of Technology, Dayton, Ohio, 1999. ISBN: 0-599-28316-5
- [12] DEB, K. *Multi-objective Optimization Using Evolutionary Algorithms*. John Wiley & Sons, 2001. ISBN: 047187339X
- [13] MAREK, M., RAID, Z., KADLEC, P. Synthesis of electromagnetic equivalents of composite sheets by multi-objective optimization of anisotropic band-stop filters. In *Proceedings of the 12th European Conference on Antennas and Propagation*. London (UK), 2018, 4 p. DOI: 10.1049/cp.2018.1094

## About the Authors ...

**Martin MAREK** was born in Cesky Krumlov, Czech Republic in 1991. He received his B.Sc. and M.Sc. degrees in Electronics and Communication in 2014 and 2016, respectively, both from the Brno University of Technology, Brno, Czech Republic. Currently, he is working towards Ph.D. degree in Electronics and Communication with the Dept. of Radio Electronics, Brno University of Technology. His research interests include multi-objective evolutionary algorithms for the optimization of the electromagnetic components.

**Petr KADLEC** was born in Brno, Czech Republic in 1985. He received his B.Sc., M.Sc. and Ph.D. degrees in Electronics and Communication in 2007, 2009 and 2012, respectively, all from the Brno University of Technology, Brno, Czech Republic. Currently, he is with the Dept. of Radio Electronics, Brno University of Technology, as a researcher. His research interests include numerical methods for electromagnetic field computations and evolutionary algorithms for the optimization of the electromagnetic components.

## Appendix A: NSGA-II Results

Controlling parameters of the NSGA-II algorithm were set as follows: the probability of crossover  $P_C = 0.9$ , the probability of mutation  $P_M = 0.7$  and a binary precision of each decision variable depends on a number of discrete samples of each decision variable ( $BP = 4$  for 11 discrete samples,  $BP = 7$  for 101 discrete samples, etc.). Although there are 16 possible combinations for  $BP = 4$  for each decision variables, the NSGA-II algorithm was set so that only the defined number of discrete positions could be reached.

The number of the discrete samples of each decision variable is analogous to the setting of MOPSO with the discrete decision space. Therefore, each decision variable was sampled by 11, 101, 1001, and  $2^{20}$  points. Note that  $2^{20}$  discrete points per decision variable represents here a continuous-like decision space.

A number of agents and a number of iterations used in the NSGA-II simulation was identical to the MOPSO algorithm setting (100 agents and 100 iterations). A number of repetitions of all simulations was 10 in this case.

Delay [ms]	Tolerance	MOFON	MOKUR	MOPOL	MOZDT1	MOZDT4
0	0	0.96	1.07	0.92	1.33	1.07
1	0	11.31	11.42	11.26	11.56	11.42
10	0	105.11	105.50	104.93	105.52	105.27
0	0.001	3.38	3.36	2.66	21.12	4.49
1	0.001	13.08	11.35	9.01	31.54	13.52
10	0.001	97.28	83.24	66.37	123.09	94.76
0	0.01	2.57	2.30	1.83	20.77	4.08
1	0.01	6.96	5.41	3.84	30.41	11.45
10	0.01	52.49	37.71	22.28	120.90	75.75
0	0.1	1.64	1.61	1.50	17.56	2.93
1	0.1	2.18	2.13	1.68	26.30	6.44
10	0.1	7.02	6.99	2.85	108.15	41.25

**Tab. 6.** An average computational time in seconds - NSGA-II.

Delay [ms]	Tolerance	MOFON	MOKUR	MOPOL	MOZDT1	MOZDT4
0	0.001	1253	2352	3997	533	1594
1	0.001	1298	2593	3979	509	1681
10	0.001	1219	2534	4069	576	1555
0	0.01	5631	6909	8104	571	3354
1	0.01	5741	7006	8114	589	3111
10	0.01	5360	6756	8115	628	3288
0	0.1	9501	9522	9882	1992	6634
1	0.1	9482	9524	9881	1754	6641
10	0.1	9500	9512	9881	1729	6456

**Tab. 7.** An average number of surrogate solutions - NSGA-II.

Delay [ms]	Tolerance	MOFON	MOKUR	MOPOL	MOZDT1	MOZDT4
0	0	0.003	0.007	0.027	0.195	6.537
1	0	0.002	0.007	0.022	0.183	5.457
10	0	0.002	0.007	0.018	0.182	6.078
0	0.001	0.002	0.005	0.031	0.119	5.066
1	0.001	0.002	0.005	0.014	0.120	5.907
10	0.001	0.002	0.005	0.019	0.131	3.860
0	0.01	0.002	0.064	0.024	0.087	4.359
1	0.01	0.002	0.065	0.024	0.086	3.625
10	0.01	0.002	0.059	0.028	0.091	3.150
0	0.1	0.102	0.017	0.288	<0.001	<0.001
1	0.1	0.102	0.017	0.288	<0.001	<0.001
10	0.1	0.102	0.017	0.289	<0.001	<0.001

**Tab. 8.** An average generational distance - NSGA-II.